DOUBLE INTEGRATION USING CUBIC SPLINES

Thesis

UNRESTRICTED

DISSERTATION:

DOUBLE INTEGRATION USING CUBIC SPLINES

by

R.WILLIAMS B.Sc.

Presented for the Bachelor of Philosophy Degree in Mathematics 31st December 1982

Date of submission : 1·1·83

Date of award : 22·7·83

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

DISSERTATION : 'DOUBLE INTEGRATION USING CUBIC SPLINES'

by R.WILLIAMS B.Sc.,M.R.I.N.

This dissertation applies the approximation methods of the cubic spline and the bicubic spline to the problem of integration in two dimensions. The dissertation is divided into six sections.

Section 1 is the Introduction and contains background material concerning the historical introduction and theory of the cubic spline together with a resumé of what it is intended to achieve in the body of the dissertation.

Section 2 contains a description of the derivation of the traditional algorithm for computing a one dimensional cubic spline approximation to a function $f(x)$ and then we integrate this cubic spline to find an approximation to the integral of $f(x)$.

In Section 3 we express the integral of a function $f(x)$ as a function of its upper limit and then fit a one dimensional cubic spline to this integral directly.

In Section 4 we apply the methods of Sections 2 and 3 to evaluate integral approximations to functions of two variables.

In Section 5 we fit a Bicubic Spline to the mesh values of a function of two variables over a rectangular mesh and integrate this bicubic spline approximation to give an approximate value of the double integral of the function.

Section 6 contains the conclusions drawn fro the preceeding computations.

CONTENTS

1. INTRODUCTION

1.1 The mathematical idea which we know as the cubic spline is developed from the desire to construct a cubic interpolating polynomial $S(x)$ to a function $f(x)$ over an interval $[a,b]$ such that $S(x)$ coincides with $f(x)$ at the points of subdivision $a=x_0, x_1, \ldots, x_{n-1}, x_n=b$ $(x_0<x_1 \ldots .<x_n)$, is continuous, with continuous first and second derivatives throughout the interval, and satisfies certain prescribed boundary conditions. In each subinterval, therefore, $S(x)$ is represented by a different cubic polynomial $s_i(x)$, for $i=1,..,n$, where the values of $s_i(x)$ and $s_{i+1}(x)$ at the point $x=x_i$, at which they are both defined, and the values of their first and second derivatives, are such that $S(x)$ satisfies the above interpolatory and continuity constraints. $S(x)$ belongs to the continuity class $C^2[a,b]$. The algorithms for finding $S(x)$ are described in Section 2.

The idea was introduced by Schoenberg[1] in 1946 in a paper entitled "Contributions to the problems of approximation of equidistant data by Analytic Functions". The term 'spline' may come from the name given to a flexible strip, used by draughtsmen, which is pinned by 'ducks' and thus constrained to form the shape of a curve passing through given points. See figure 1. Such a method would probably be used, for instance, to fit the curve of a ship's hull to its frames. By the physical nature of such a flexible strip it is not difficult to imagine that a curve so drawn will have some continuity of its derivatives and so, since the essential features of the function $S(x)$ are the continuity of its derivatives, the term 'spline' was probably adopted for the function as well from this.

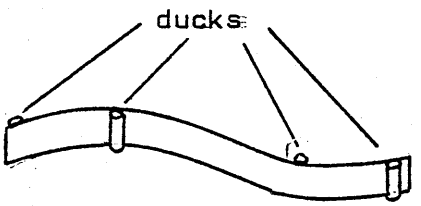


Figure 1.

The cubic spline is not the only spline function. Other polynomial splines, also used in the problem of interpolation, such as quintic splines, have been investigated and there is a theory of generalised splines. This theory is well covered in the book by Ahlberg, Nilson & Walsh[2]. Briefly, following along the lines of the definition given in the book by P.Prenter[3], we can say that associated with any interval $[x_0,x_n]$, partitioned

into a mesh by fixed points $x_1,\ldots,x_{n-1}$, there is a space of functions, S_3, belonging to $C^2[x_o,x_n]$ which reduce to a cubic polynomial in each subinterval $[x_{i-1},x_i]$ for $i=1,..,n$. In particular, there is a unique function $s(x) \in S_3$ satisfying the interpolation problem

$$s'(x_o) = f'(x_o) \qquad s'(x_n) = f'(x_n)$$
$$s(x_i) = f(x_i) \qquad i=0,..,n \qquad \ldots\ldots 1.1.1$$

and this is the cubic spline approximation to $f(x)$ in $[x_o,x_n]$. Cubic splines are therefore a space of functions, quite independent of the interpolation problem, which have an independent theoretical significance.

We can find a basis for the space S_3 of those cubic splines where values for the first derivative at the boundary of the interval are known. These are the Cardinal Splines - the $n+3$ linearly independent cubic splines which satisfy the interpolatory constraints

$$\psi_k(x_j) = \delta_{kj} \quad (j=0,..,n) \qquad \psi_k'(x_j) = 0 \quad (j=0,n)$$

for $k=0,..,n$

$$\varphi_k(x_j) = 0 \quad (j=0,..,n) \qquad \varphi_k'(x_j) = \delta_{kj} \quad (j=0,n)$$

for $k=0,n$.

The cubic spline $s(x)$ satisfying the conditions 1.1.1 can therefore be expressed as a linear combination of these cardinal splines :

$$s(x) = \sum_{j=o}^{n} \psi_j(x)\, f(x_j) + \varphi_o'(x)\, f'(x_o) + \varphi_n'(x)\, f'(x_n)$$

$\ldots\ldots$ 1.1.2

In some applications of cubic splines 1.1.2 forms the substance of an algorithm but there are more economical algorithms for the problem of interpolation.

The idea of the cubic spline can be quickly and easily generalised to the polynomial spline through a definition similar to that above given for the cubic spline :

In an interval $[x_o,x_n]$ partitioned into a mesh by fixed points $x_1,\ldots,x_{n-1}$ there is a space of functions S_m belonging to the

class $C^{m-1}[x_o,x_n]$ which reduce to a polynomial of degree m in each subinterval $[x_{i-1},x_i]$, for i=1,..,n . These functions are the polynomial splines of degree m .

We will not discuss the application of generalised splines to the problem of interpolation and approximation; we confine ourselves, in this dissertation, to the application of the cubic spline and its two dimensional counterpart, the bicubic spline, to problems of integration, particularly in two dimensions.

Boundary conditions are an important consideration. In 1.1.1 we take it that $f'(x_o)$ and $f'(x_n)$ are known, but, if this is not the case, then some other equations of constraint must be considered because the algorithms require it. A common practice, when $f'(x_o)$ and $f'(x_n)$ are not known, is to assume the conditions $S''(x_o) = 0$, $S''(x_n) = 0$ because, out of all the functions in $C^2[a,b]$ which interpolate to the function f(x) at the mesh points $a=x_o$, x_1,.. ..,x_{n-1}, $x_n=b$ then S(x) minimises the integral

$$\int_a^b |f''(x)|^2 \, dx \quad .$$

The spline is then known as the 'Natural Spline' . There is, however, a reduction in accuracy by using the natural spline to approximate to f(x). The accuracy can be rescued somewhat by using the Lagrangian Spline, so called, in this case, because we estimate $f'(x_o)$ and $f'(x_n)$ by fitting a Lagrange polynomial to the first three or four ordinates and last three or four ordinates in the interval and differentiating these approximations. The details are shown in section 3 .

1.2 In section 2 of this dissertation we describe how to find S(x) which approximates to the function f(x) on the mesh $x_o<x_1<$ .. .. $<x_n$. This mesh need not be a regular mesh even though the regular mesh is used in the examples. First we define S(x) by an explicit polynomial form in each subinterval $[x_{i-1},x_i]$ for i=1,..,n but, for purposes of computation, the more widely used algorithm expresses S(x) in terms of 'moments', M_i, where $M_i = S''(x_i)$. This 'moment' algorithm results in a tridiagonal system of linear equations which is easier to solve than the system which results when finding the coefficients of the explicit polynomial elements of S(x) even though this latter system is also sparse. S(x), expressed in terms of moments, is easy to integrate. The result is the Trapezoidal Rule with correction terms as is shown by Davis & Rabinowitz[4]. Some results of using this formula are shown

at the end of section 2.

1.3 In section 3 we introduce a variation in the method of computing the cubic spline approximation in order to develop the cubic spline approximation to the integral of f(x) directly. We fit the cubic spline S(x) to the function F(x), which is the definite integral of f(x) over the interval $[x_o,x]$ so that, if $s_i(x)$ is the representation of S(x) in the interval $[x_{i-1},x_i]$, then

$$s_i(x) \approx \int_{x_o}^{x} f(x)\,dx \qquad \text{for } x \in [x_{i-1},x_i] .$$

We use the known values $S(x_i) = F'(x_i) = f(x_i)$ for interpolation at the mesh points, and, as constraint conditions, we use the continuity of S(x) and S"(x) .

As before, we need two further boundary conditions to apply the algorithms, but, this time, we attach an additional importance to the boundary conditions because they determine the uniqueness of the cubic spline spline solution to this interpolation problem. As it happens this uniqueness problem is immediately solved because, by definition, $F(x_o) = 0$. After that we can find another condition to suit the algorithm in use. Finding S(x) in explicit polynomial form will require a boundary condition at $x=x_n$ but if we find S(x) in terms of moments then a further condition at $x=x_o$ is requred and this is found using either the Natural or Lagrangian splines as in section 2.

Some results found by using this method are shown at the end of section 3.

1.4 Section 4 deals with the application of these one dimensional cubic splines to the problem of integration in two dimensions. Both methods are efficient but there is a noticeable reduction in computing time using the 'direct' method of section 3. This remark applies equally well, in fact, for the applications of the two methods to one dimensional integrals.

1.5 In section 5 we study the application of the bicubic spline to the rectangular region R where we approximate to the function f(x,y) of two variables. There are two particular papers which refer to the application of the bicubic spline to the problem of interpolation and approximation; those by Birkhoff & Garabedian[5] and Carl de Boor[6]. These papers are the basis of the work here.

In a mesh rectangle r_{ij} of the region R, the bicubic spline can be expressed in an expanded form

$$s_{ij}(x,y) = \sum_{u=0}^{3} \sum_{v=0}^{3} (c_{ij})_{uv} x^u y^v \qquad 1.5.1$$

or in a product form

$$s_{ij}(x,y) = \left(\sum_{u=0}^{3} a_u x^u \right) \left(\sum_{v=0}^{3} b_v y^v \right) \qquad 1.5.2$$

Each form lends itself to different algorithms but in this dissertation we restrict ourselves to a computation algorithm based on the expanded form, 1.5.1, of the bicubic spline. We apply the algorithm as set out in the paper by Carl de Boor, and compute the values of the first partial and second mixed partial derivatives at the corners of the mesh rectangles of R from one dimensional cubic splines fitted to the function values, or the computed first derivative values, as the case may be, along the mesh lines in each direction. The explicit bicubic polynomial coefficients, $(c_{ij})_{uv}$, can then be computed in each mesh rectangle and the bicubic polynomial can be integrated, a sum being taken over all the mesh rectangles in the region R.

1.6 The computations are all performed on a TRS 80 microcomputer which works to six significant figures in single precision arithmetic and to twelve in bouble precision. We have only made use of the single precision feature. In some of the examples, therefore, it is possible that the truncation error is 'blurred' by round off error.

2. THE CUBIC SPLINE APPROXIMATION TO THE FUNCTION f(x)

2.1 Representation of the Cubic Spline in explicit polynomial form

Consider a function $f(x) \in C^4$ defined on the interval $[x_o, x_n]$. The interval is subdivided by points $x_1, x_2, \ldots, x_{n-1}$. We can approximate to f(x) using a cubic spline approximation S(x) which is defined in the following manner. S(x) is continuous and has continuous first and second derivatives throughout $[x_o, x_n]$. At each point x_i for i=0,..,n :

$$S(x_i) = f(x_i)$$

and, at the boundary points $x=x_o$ and $x=x_n$, we have

$$S'(x_o) = f'(x_o)$$

and

$$S'(x_n) = f'(x_n) \quad .$$

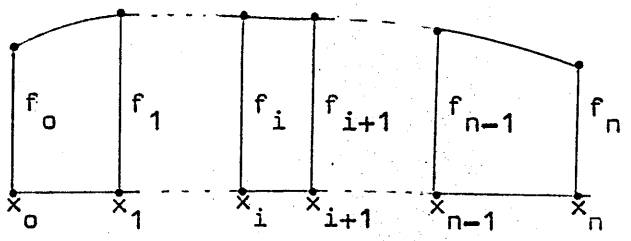


Figure 2.

In each subinterval $[x_{i-1}, x_i]$ for i=1,..,n , S(x) reduces to the cubic polynomial $s_i(x)$ and so S(x) is a piecewise continuous cubic polynomial throughout the interval $[x_o, x_n]$.

We can illustrate the existence of S(x) in its explicit polynomial form by the following algorithm:

At each point $x=x_i$ for i=1,..,n-1 , we have, since S(x) is continuous,

$$s_i(x_i) = f(x_i)$$

and

$$s_{i+1}(x_i) = f(x_i) \qquad \ldots\ldots \quad 2.1.1$$

and, since S'(x) and S"(x) are also continuous, by definition, then

$$s_i'(x_i) = s_{i+1}'(x_i)$$

and

$$s_i''(x_i) = s_{i+1}''(x_i) \qquad \ldots\ldots \quad 2.1.2$$

On the boundary, assuming f'(x) is known,

$$s_1(x_o) = f(x_o)$$

$$s_1'(x_o) = f'(x_o) \qquad \ldots\ldots \quad 2.1.3\ a$$

$$s_n(x_n) = f(x_n)$$
$$s_n'(x_n) = f'(x_n) \qquad \ldots\ldots \quad 2.1.3\ b$$

If we express the $s_i(x)$ in explicit polynomial form

$$s_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$$

then the sets of conditions 2.1.1, 2.1.2, and 2.1.3 provide a system of 4n linear equations from which to compute the 4n coefficients (a_i, b_i, c_i, d_i) . It is then an easy task to compute the integral approximation to f(x) using S(x) . We find

$$\int_{x_o}^{x_n} f(x)\,dx \sim \int_{x_o}^{x_n} S(x) = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} s_i(x)\,dx \quad .$$

2.2 Representation of the Cubic Spline in terms of 'Moments'.
The explicit representation of S(x) in terms of the $s_i(x)$ is not efficient for computation. A widely used method expresses S(x) in terms of 'Moments', M_i, which are the values of S"(x) at the mesh points $x=x_i$, i=0,...,n . This method is described by Ahlberg, Nilson & Walsh[2]. We begin by expressing S"(x) as a linear function in each subinterval $[x_{i-1}, x_i]$:

$$S''(x) = \frac{M_i}{h_i}(x-x_{i-1}) + \frac{M_{i-1}}{h_i}(x_i-x) \qquad \ldots\ldots\ 2.2.1$$

where $h_i = x_i - x_{i-1}$.

If we integrate equation 2.2.1 twice and evaluate the constants of integration by collocation at the end points of the subintervals, (i.e. by making use of the conditions $S(x_i) = f(x_i)$ and $S(x_{i-1}) = f(x_{i-1})$) then, in $[x_{i-1}, x_i]$:

$$S(x) = \frac{M_{i-1}}{6h_i}(x_i-x)^3 + \frac{M_i}{6h_i}(x-x_{i-1})^3 + \left[f_{i-1} - \frac{M_{i-1}h_i^2}{6}\right]\frac{(x_i-x)}{h_i} + \left[f_i - \frac{M_i h_i^2}{6}\right]\frac{(x-x_{i-1})}{h_i} \qquad \ldots\ldots\ 2.2.2$$

and, by differentiating 2.2.2 , we find

$$S'(x) = -\frac{M_{i-1}}{2h_i}(x_i-x)^2 + \frac{M_i}{2h_i}(x-x_{i-1})^2 + \frac{(f_i - f_{i-1})}{h_i} - \frac{(M_i - M_{i-1})}{6}h_i \qquad \ldots\ 2.2.3$$

where $f_i = f(x_i)$.

Since $S'(x)$ is continuous, then, at internal mesh points $x=x_i$, $i=1,..,n-1$, we can use the equality of left and right hand derivatives :

$$S'(x_i-) = \left(\frac{M_{i-1} + 2M_i}{6}\right) h_i + \left(\frac{f_i - f_{i-1}}{h_i}\right)$$

and $$S'(x_i+) = -\left(\frac{2M_i + M_{i+1}}{6}\right)h_{i+1} + \left(\frac{f_{i+1} - f_i}{h_{i+1}}\right)$$

which gives

$$\frac{h_i M_{i-1}}{6} + \left(\frac{h_i + h_{i+1}}{3}\right) M_i + \frac{h_{i+1} M_{i+1}}{6} = \left(\frac{f_{i+1} - f_i}{h_{i+1}}\right) - \left(\frac{f_i - f_{i-1}}{h_i}\right)$$

for $i=1,...,n-1$.

In order to provide a set of $n+1$ linear equations from which to compute the $n+1$ moments M_i, we require two more equations. The last two equations we find from the boundary conditions.
If f'_o and f'_n are known then

$$\frac{M_o}{3} h_1 + \frac{M_1}{6} h_1 = \frac{f_1 - f_o}{h_1} - f'_o$$

and $$\frac{M_{n-1}}{6} h_n + \frac{M_n}{3} h_n = f'_n - \frac{f_n - f_{n-1}}{h_n} \quad .$$

To compute an approximation to the integral of $f(x)$ over the interval $[x_o, x_n]$ then

$$\int_{x_o}^{x_n} f(x)\, dx \sim \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} S(x)\, dx$$

where $S(x)$ is expressed by 2.2.2 for each i .

As is shown by Davis and Rabinowitz[4], this is found to be

$$\int_{x_o}^{x_n} f(x)\, dx \sim \sum_{i=1}^{n} \tfrac{1}{2}(f_{i-1} + f_i)h_i - \sum_{i=1}^{n} \left(\frac{M_{i-1} + M_i}{24}\right) h_i^3 \qquad \text{.....} \quad 2.2.4$$

which is the Trapezoidal Rule with correction terms.

2.3 Computer Programme.

The following computer programme is written in BASIC to evaluate the integral

$$\int_{x_0}^{x_n} f(x)\,dx$$

using the method of this section.

```
 10 INPUT N,XO,XN:DIM A(N,2),B(N),X(N),F(N),M(N)
```

X(I) is x_i, F(I) is $f(x_i)$. A, B and M form the linear system AM=B

```
 15 X(0)=XO:X(N)=XN
 20 H=(X(N)-X(0))/N:FOR I=0 TO N:X(I)=X(0)+I*H:XI=X(I)
 30 GOSUB 500:F(I)=FX:NEXT I:A(0,1)=H/3:A(0,2)=H/6
 40 A(N,0)=H/6:A(N,1)=H/3:XI=X(0):GOSUB600
 50 B(0)=(F(1)-F(0))/H-FD:XI=X(N):GOSUB600
 60 B(N)=FD-(F(N)-F(N-1))/H:FOR I=1 TO N-1
 70 A(I,0)=H/6:A(I,1)=(2*H)/3:A(I,2)=H/6
 80 B(I)=(F(I+1)-2*F(I)+F(I-1))/H:NEXT I
 90 FOR I=1TO N:A(I,1)=A(I,1)-A(I-1,2)*A(I,0)/A(I-1,1)
100 B(I)=B(I)-B(I-1)*A(I,0)/A(I-1,1):A(I,0)=0:NEXT I
110 M(N)=B(N)/A(N,1):K=N-1:FOR I=K TO 0 STEP-1
120 M(I)=(B(I)-A(I,2)*M(I+1))/A(I,1):NEXT I
130 FOR I=1 TO N
140 S=S+(F(I)+F(I-1))*H/2-(M(I)+M(I-1))*(H^3)/24
150 NEXT I:PRINT S:END
500 FX   (Function f(x))
600 FD   (Derivative f'(x))
```

2.4 Numerical Examples.

Function	interval	mesh size	numerical results	true value	error
sin x	$[0,\pi]$	$\pi/20$	2.000000	2.000000	0.000000
cos x	$[0,\pi]$	$\pi/20$	0.000003	0.000000	0.000003
tan x	$[0,\frac{1}{4}\pi]$	$\pi/20$	0.346562	0.346574	0.000012
1/x	$[1,2]$	0.1	0.693146	0.693147	0.000001

2.5 Error bounds

In 2.2 we found that

$$\int_{x_o}^{x_n} S(x)\,dx = \sum_{i=1}^{n} \tfrac{1}{2}(f_{i-1}+f_i)h_i - \sum_{i=1}^{n} \left(\frac{M_{i-1}+M_i}{24}\right)h_i^3$$

where $S(x)$ is the cubic spline approximation to $f(x)$ on $[x_o,x_n]$ and exact derivative boundary conditions are assumed known. On a regular mesh where the mesh length is a constant h, this becomes

$$\int_{x_o}^{x_n} S(x)\,dx = h\sum_{i=1}^{n} \tfrac{1}{2}(f_{i-1}+f_i) - \frac{h^3}{24}\sum_{i=1}^{n}(M_{i-1}+M_i) \qquad \ldots\ldots\ 2.5.1$$

We also know that

$$\int_{x_o}^{x_n} S''(x)\,dx = S'(x_n) - S'(x_o)$$

and, since $S''(x)$ is a piecewise linear function of x on $[x_o,x_n]$, it is integrated exactly by the Trapezoidal Rule, hence

$$\int_{x_o}^{x_n} S''(x)\,dx = h\sum_{i=1}^{n} \tfrac{1}{2}(M_{i-1}+M_i) = S'(x_n) - S'(x_o)\ .$$

Given that $S'(x_n) = f'(x_n)$ and $S'(x_o) = f'(x_o)$ and that $f'(x_n)$ and $f'(x_o)$ are known exactly, 2.5.1 can be written

$$\int_{x_o}^{x_n} S(x)\,dx = h\sum_{i=1}^{n} \tfrac{1}{2}(f_{i-1}+f_i) - \frac{h^2}{12}(f'_n - f'_o) \qquad \ldots\ldots\ 2.5.2$$

If we use the Euler - Maclaurin formula to evaluate the integral of f over the interval $[x_o,x_n]$ we find that

$$\int_{x_o}^{x_n} f(x)\,dx = h\sum_{i=1}^{n} \tfrac{1}{2}(f_{i-1}+f_i) - \frac{h^2}{12}(f'_n - f'_o) + \frac{h^4}{720}(f^{iii}_n - f^{iii}_o) + O(h^6) \quad \ldots\ 2.5.3$$

Subtracting 2.5.2 from 2.5.3 gives

$$\int_{x_o}^{x_n} (f-S)\,dx = \frac{h^4}{720}(f^{iii}_n - f^{iii}_o) + O(h^6)\ .$$

Using the Mean Value Theorem for integrals we find that, for some ξ such that $x_o \leqslant \xi \leqslant x_n$:

$$\int_{x_o}^{x_n} (f-S)\,dx = \frac{nh^5}{720} f^{iv}(\xi) + O(h^6) \quad \ldots\ldots 2.5.4$$

The error term for the Euler Maclaurin formula is given as

$$\frac{nh^5}{720} f^{iv}(\eta)$$

for some η such that $x_o \leqslant \eta \leqslant x_n$.

If we take $\|f^{iv}\|$ as the maximum absolute value of $f^{iv}(x)$ in $[x_o, x_n]$ then a bound for

$$\left| \int_{x_o}^{x_n} (f-S)\,dx \right|$$

will be $$\frac{nh^5}{720} \|f^{iv}\| \quad \ldots\ldots 2.5.5$$

The source of error analysis for the one dimensional cubic spline most often quoted is in the paper "On Error Bounds for Spline Interpolation" by C.A.Hall[7], in which, for a cubic spline S, with evenly spaced knots in an interval a,b, approximating to a function $f \in C^4\ a,b$, with given derivative boundary conditions which are exact, the error bound, in terms of the maximum norm, is given by

$$|f - S| \leqslant \frac{5h^4}{384} \|f^{iv}\| \quad \ldots\ldots 2.5.6$$

We find then that

$$\left| \int_{x_o}^{x_n} f(x)\,dx - \int_{x_o}^{x_n} S(x)\,dx \right| \leqslant (x_n - x_o)\frac{5}{384} h^4 \|f^{iv}\|$$

where the norm is the absolute maximum value in $[x_o, x_n]$. Hence

$$\left| \int_{x_o}^{x_n} (f-S)\,dx \right| \leqslant \frac{5nh^5}{384} \|f^{iv}\| \quad \ldots\ldots 2.5.7$$

2.6 The table below shows the results of 2.4 in relation to the error bounds 2.5.5 and 2.5.7

Function	Actual errors	f^{iv}	Error Bounds 2.5.5	2.5.7
sin x	0.000000	1.0	0.000003	0.000024
cos x	0.000003	1.0	0.000003	0.000024
tan x	0.000012	80.0	0.000053	0.000498
1/x	0.000001	24.0	0.000003	0.000031

2.7 If the interpolation problem is presented without given derivative boundary conditions (which is the more recognisable situation in practice) then it becomes necessary to "invent" boundary conditions, or equations of constraint close to the boundary, such as 2.7.1 below, which will make up the requisite number of linear equations from which to compute the cubic spline. The first possibility, frequently quoted as an answer in the above circumstances, is to assume the so called "free end" conditions: $s''(x_o) = o$ and $s''(x_n) = o$ .These conditions are very easy to apply but, unfortunately, they lead to an error bound of order h^2 in the intervals close to the boundary. By the nature of the cubic spline approximation this low order error is 'smoothed out' in the interior of the interval when the number of points of subdivision of the interval is sufficiently large.
Another set of conditions (which are, in effect, additional interpolatory constraints) are the assumed equations (Curtis[8])

$$(s - f)_{x=\frac{1}{2}(x_1+x_o)} = (s - f)_{x=\frac{1}{2}(x_2+x_1)}$$

$$(s - f)_{x=\frac{1}{2}(x_{n-1}+x_{n-2})} = (s-f)_{x=\frac{1}{2}(x_n+x_{n-1})}$$

..... 2.7.1

These equations are useful if f can be evaluated at these half interval points. If, however, the only information on f is a set of data points then this will not, in general, be possible. In this case, then, we can use the Lagrangian Spline. To maintain error bounds of order h^4 at the boundary as well as the interior points, we should fit a cubic Lagrange polynomial to the first four (and the last four) ordinates in the interval and then differentiate this polynomial to find approximate values for the derivatives at the boundary. The error bounds for the Lagrangian Splines (S_L - Splines) in approximation to functions are discussed (in a very general setting) in the paper by Swartz and Varga[9].

3. DIRECT CUBIC SPLINE APPROXIMATION TO THE INTEGRAL OF A FUNCTION

3.1 Let $$F(x) = \int_a^x f(x)\,dx \qquad \ldots\ldots \quad 3.1.1$$

where $f(x)$ is defined and continuous in the interval $[a,b]$ and $f \in C^4[a,b]$, then

$$F'(x) = f(x) \quad \text{in } [a,b] .$$

Let us subdivide the interval $[a,b]$ into a mesh by points $x_1,\ldots,x_{n-1}$ such that

$$a = x_o \leqslant x_1 \quad \ldots\ldots \leqslant x_{n-1} \leqslant x_n = b$$

We now approximate to $F(x)$ using the cubic polynomial spline $S(x)$ where $S(x)$ is a piecewise cubic polynomial which is continuous and has continuous first and second derivatives throughout $[a,b]$, and such that $S'(x) = f(x)$ at the mesh points. In the subinterval $[x_{i-1},x_i]$, $S(x)$ coincides with the cubic polynomial $s_i(x)$ where

$$s_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \ .$$

Since there are n such pieces $s_i(x)$, $i=1,\ldots,n$, we could determine the 4n coefficients (a_i,b_i,c_i,d_i) by setting up a system of 4n linear equations using the above constraints. Since, at $x=x_i$, $S(x)$ is continuous then we can match the adjacent elements of $S(x)$, i.e.

$$s_i(x_i) = s_{i+1}(x_i) \qquad \text{for } i=1,\ldots,n-1$$

We can fit $s_i'(x_i)$ and $s_{i+1}'(x_i)$ to $f(x_i)$, i.e.

$$s_i'(x_i) = f(x_i)$$

and $$s_{i+1}'(x_i) = f(x_i) \qquad \text{for } i=1,\ldots,n-1$$

and, since $S''(x)$ is continuous throughout $[a,b]$ we can match the adjacent elements $s_i''(x)$ and $s''_{i+1}(x)$ of $S''(x)$ at $x=x_i$, i.e

$$s_i''(x_i) = s_{i+1}''(x_i) \qquad \text{for } i=1,\ldots,n-1 \ .$$

We can make up the remaining 4 equations from the boundary values, but care must be taken to include at least one value of F because, in this case, the uniqueness of the solution to the problem depends upon it. We can achieve this by using the obvious condition $F(x_o) = 0$ which follows from the definition of $F(x)$. Hence, at $x=x_o$ we use the conditions

$$s_1'(x_o) = f(x_o) \quad \text{and} \quad s_1(x_o) = F(x_o) = 0 \ .$$

At $x=x_n$ we can use the conditions

$$s_n'(x_n) = f(x_n)$$

and $$s_n''(x_n) = f'(x_n)$$ if this value is known .

3.2 The method of expressing $S(x)$ in explicit polynomial form in each subinterval is not very efficient for computation though it does show the existence and uniqueness of $S(x)$ effectively. In practice it is preferable to express $S''(x)$ in terms of 'moments' M_i , as we did in the previous section, and integrate $S''(x)$ twice to find $S(x)$. In $[x_{i-1},x_i]$, for $i=1,..,n$, we find

$$S''(x) = \frac{M_{i-1}}{h_i}(x_i-x) + \frac{M_i}{h_i}(x-x_{i-1}) \qquad \ldots\ldots 3.2.1$$

where $h_i = x_i-x_{i-1}$.

Integrating 3.2.1 gives

$$S'(x) = -\frac{M_{i-1}}{2h_i}(x_i-x)^2 + \frac{M_i}{2h_i}(x-x_{i-1})^2 + C_{1i}$$

where C_{1i} is the constant of integration. We determine C_{1i} by equating $S'(x)$ to $f(x)$ at either $x=x_{i-1}$ or $x=x_i$. Choosing $x=x_i$ we find

$$S'(x_{i-1}) = -\frac{M_{i-1}}{2h_i}(x_i-x_{i-1})^2 + C_{1i} = f_{i-1}$$

where $f_i = f(x_i)$. This gives us C_{1i} :

$$C_{1i} = f_{i-1} + \frac{M_{i-1}}{2}h_i \quad .$$

In the interval $[x_{i-1},x_i]$ we therefore find $S'(x)$ in the form

$$S'(x) = \frac{M_i}{2h_i}(x-x_{i-1})^2 + \frac{M_{i-1}}{2h_i}\left[h_i^2 - (x_i-x)^2\right] + f_{i-1}$$

$\ldots\ldots$ 3.2.2

And, in particular, at $x=x_i$ we find that

$$S'(x_i) = f_i = \tfrac{1}{2}M_ih_i + \tfrac{1}{2}M_{i-1}h_i + f_{i-1} \qquad \text{so that}$$

$$M_i + M_{i-1} = 2(f_i - f_{i-1})/h_i \quad .$$

Form 3.2.1 $M_o = S''(x_o)$ and we also know that $S''(x_o) = f_o'$.

We can therefore determine the M_i from the scheme

$$M_o = f'_o$$
$$M_i = 2(f_i - f_{i-1})/h_i - M_{i-1} \quad i=1,\ldots,n \quad .$$

If the value of f'_o is not known then we are forced to consider some alternatives. We can choose $S(x)$ to be the 'Natural' spline and set $S''(x_o) = 0$, but this reduces the accuracy of the approximation in the intervals close to the boundary. The best alternative, in this situation, is to make $S(x)$ into a Lagrangian Spline (see Prenter[3] page 83) where we fit a Lagrange polynomial $L(x)$ to the four ordinates f_o, f_1, f_2, f_3 at the points $x=x_o$, $x=x_1$, $x=x_2$ and $x=x_3$, respectively, so that, in the interval $[x_o, x_3]$:

$$f(x) \approx L(x)$$

and $$f'(x) \approx L'(x) \quad .$$

$L(x)$ is a cubic polynomial whose error is of the same order as the cubic spline.

We now find the general form of $S(x)$ in terms of the M_i in the interval $[x_{i-1}, x_i]$ by integrating 3.2.2 .

$$S(x) = \frac{M_i}{6h_i}(x-x_{i-1})^3 + \frac{M_{i-1}}{2h_i}\left[h_i^2 x + \tfrac{1}{3}(x_i-x)^3\right] + f_{i-1}x + C_{2i} \qquad \ldots\ldots \; 3.2.3$$

We cannot determine the constants of integration, C_{2i}, directly. We do so by first determining C_{21} and then the remaining constants successively. In the interval $[x_o, x_1]$:

$$S(x) = \frac{M_1}{6h_1}(x-x_o)^3 + \frac{M_o}{2h_1}\left[h_1^2 x + \tfrac{1}{3}(x_1-x)^3\right] + f_o x + C_{21} \quad .$$

At $x=x_o$ $S(x_o) = F(x_o) = 0$ therefore

$$C_{21} = -\frac{M_o}{2h_1}(h_1^2 x_o + \tfrac{1}{3}h_1^3) - f_o x_o \quad .$$

This determines $S(x)$ in the interval $[x_o, x_1]$ and from it we can find a value for $S(x_1)$. We apply 3.2.3 again, this time in the interval $[x_1, x_2]$ and use the value of $S(x_1)$ to determine C_{22} , then determine $S(x_2)$ and so on until $i=n$. Ultimately we find

$$S(x_n) \sim \int_a^b f(x)\,dx \quad .$$

3.3 The Computer Programme

The following computer programme is written in BASIC to evaluate the integral

$$\int_{x_o}^{x_n} f(x)\,dx$$

using the method of this section. In this instance the boundary conditions are not programmed but a value for M_o is part of the input.

```
10 INPUT "N= ";N
```

N is the number of intervals in the mesh

```
15 DIM S(N),C(N),F(N),X(N),M(N)
```

S(I) is the value of $F(x_i)$;C(I) contains the constants of integration C_{2i} ;F(I) is the value of $f(x_i)$;X(I) is the value x_i; M(I) is M_i .

```
20 INPUT "VALUES FOR M(0),XO,H";M(0),XO,H
```

XO is the value of x_o ; H is the mesh size for a regular mesh.

```
30 FOR I=0 TO N:X(I)=XO+I*H:NEXT I:J=N-1
40 FOR I=0 TO J:XO=X(I):GOSUB 500:F(I)=FX:XO+X(I+1):GOSUB 500
50 F(I+1)=FX:M(I+1)=2*(F(I+1)-F(I))/H-M(I):NEXT I
```

This has determined the values of x_i, f_i and M_i for i=0,..,n

```
60 S(0)=0:FOR I=1 TO N
70 C(I)=S(I-1)-M(I-1)*(H*X(I-1)+((H^2)/3))/2-F(I-1)*X(I-1)
```

This has determined C_{2i} .

```
80 S(I)=M(I)*(H^2)/6+M(I-1)*H*X(I)/2+F(I-1)*X(I)+C(I):NEXT I
90 PRINT "THE INTEGRAL OF F(X) IS S(N)= ";S(N):END
```

S(N) gives the integral approximation.

```
500 FX  (Function subroutine) .
```

3.4 Numerical Results

Applying the above computer programme the following results were obtained using the various functions on a regular mesh.

Function	Interval	mesh size	numerical results	True value	Absolute error
cos x	$[0,\frac{1}{2}\pi]$	$\pi/20$	1.00000	1.00000	0.00000
sin x	$[0,\frac{1}{2}\pi]$	$\pi/20$	1.00001	1.00000	0.00001
tan x	$[0,\frac{1}{4}\pi]$	$\pi/40$	0.346578	0.356574	0.000004
1/x	[1,2]	0.1	0.693150	0.693147	0.000003
erf(x)	[0,1]	0.1	0.842703	0.842701	0.000002

3.5 In section 3.2 we determined the 'moments' of the direct cubic spline approximation to the integral of f(x) by using the scheme

$$M_o = f'_o$$

$$M_i + M_{i-1} = 2(f_i - f_{i-1})/h \qquad i=1,..,n \qquad ....3.5.1$$

Setting $x=x_i$ and $x=x_{i-1}$ in 3.2.3 :

$$S(x_i) = \frac{M_i}{6} h^2 + \frac{M_{i-1}}{2} hx_i + x_i f_{i-1} + C_{2i}$$

$$S(x_{i-1}) = \frac{M_{i-1}}{2} hx_{i-1} + \frac{M_{i-1}}{6} h^2 + x_{i-1} f_{i-1} + C_{2i}$$

we find

$$S(x_i) - S(x_{i-1}) = \frac{h^2}{6}(M_i - M_{i-1}) + \frac{h}{2} M_{i-1}(x_i - x_{i-1}) + (x_i - x_{i-1}) f_{i-1}$$

$$= \frac{h^2}{6}(M_i - M_{i-1}) + \frac{h^2}{2} M_{i-1} + hf_{i-1}$$

$$= \frac{h^2}{6}(M_i + 2M_{i-1}) + hf_{i-1}$$

$$= \frac{h^2}{6} \frac{3}{2}(M_i + M_{i-1}) - \tfrac{1}{2}(M_i - M_{i-1}) + hf_{i-1}$$

$$= \frac{h^2}{6} \frac{3(f_i - f_{i-1})}{h} - \tfrac{1}{2}(M_i - M_{i-1}) + hf_{i-1}$$

$$= \frac{h}{2}(f_i + f_{i-1}) - \frac{h^2}{12}(M_i - M_{i-1}) \quad .$$

Now $$S(x_n) - S(x_o) = \sum_{i=1}^{n} (S(x_i) - S(x_{i-1}))$$

$$= \sum_{i=1}^{n} \frac{h}{2}(f_i + f_{i-1}) - \frac{h^2}{12} \sum_{i=1}^{n} (M_i - M_{i-1})$$

$$= \frac{h}{2} \sum_{i=1}^{n} (f_i + f_{i-1}) - \frac{h^2}{12}(M_n - M_o)$$

Since $M_n = f'_n$ and $M_o = f'_o$ we have

$$S(x_n) - S(x_o) = \frac{h}{2} \sum_{i=1}^{n} (f_i + f_{i-1}) - \frac{h^2}{12}(f'_n - f'_o) \qquad ..... 3.5.2$$

3.5.2 is exactly the same result as we obtained in section 2.5

equation 2.5.2 and shows that, for a regular mesh anyway, the methods of sections 2 and 3 are equivalent and therefore the same error bounds apply.

3.6 This method of section 3 - the direct cubic spline approximation to the integral of a function - proves to be very efficient. The algorithm is compact and the computer programme is short. The computation can also be executed using a programmable calculator or even an ordinary calculator which has multiple memories.

The algorithm is also useful for adaptive integration. We are not restricted by the method of computation to a particular number of ordinates in advance. For example, the integral

$$I = \int_0^1 x^x \, dx$$

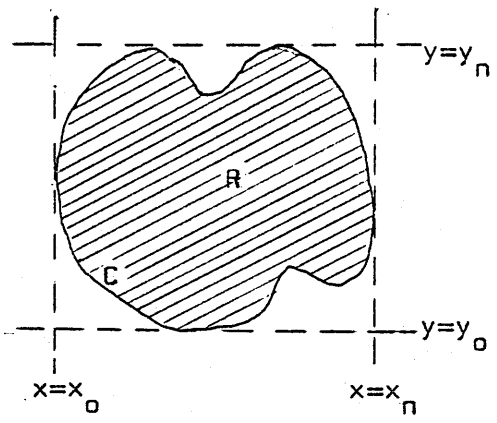
was computed by setting $x_o=1$ and using nineteen steps at constant interval $h_i = -0.05$ for $i=1,..,19$ then continuing the computation with intervals $h_{i+19} = -0.05 \times (0.5)^i$ for $i=1,2,....$ until two successive approximations differ by less than a preassigned small value. Four decimal place accuracy was achieved very quickly.

4. COMPUTATION OF DOUBLE INTEGRALS USING ONE DIMENSIONAL CUBIC SPLINES .

4.1 Consider a function $f(x,y)$ of two variables x and y, defined in a region R of the x-y plane bounded by a closed curve C . $f(x,y)$ must belong to the space of continuous functions $\mathcal{C}_2^4[R]$ by which we mean the space of functions in two variables which are continuous and have continuous partial and mixed partial derivatives up to and including order 4 .

Let R be contained in the rectangle $x_o \leqslant x \leqslant x_n$, $y_o \leqslant y \leqslant y_n$. where x_o and x_n are the extreme values of x on C and y_o and y_n are the extreme values of y on C . We wish to find a value for the integral of $f(x,y)$ over R.

Let the closed curve C which bounds the region R be such that any line $x=a$ where $x_o \leqslant a \leqslant x_n$ intersects C in at most two points. This means that we must avoid the situation where the curve C is re-entrant in such a manner that any line $x=a$ intersects the curve C in more than two points. It will not matter that any line $y=b$ $(y_o \leqslant y \leqslant y_n)$ intersects C in more than two points.



Let the interval $[x_o, x_n]$ be subdivided by points $x_1, x_2, \ldots, x_{n-1}$ such that

$$x_o < x_1 < x_2 \ldots\ldots < x_{n-1} < x_n \ .$$

Let the line $x=x_i$ intersect the curve C at points where $y=y_{io}$ and $y=y_{in}$ $(y_{io} \leqslant y_{in})$ and let the interval $[y_{io}, y_{in}]$ along the line $x=x_i$ be subdivided by points where $y=y_{i1}, \ldots, y=y_{in-1}$ such that

$$y_{io} < y_{i1} < \ldots\ldots\ldots < y_{in-1} < y_{in} \ .$$

4.2 Fitting the cubic spline of section 2

Along the line $x=x_i$ we will fit a one dimensional cubic spline $S_i(y)$ to the values of the function $f(x_i,y)$ at the points where $y=y_{io}$, $y=y_{i1}, \ldots, y=y_{in}$ using the boundary conditions

$$S_i'(y_{io}) = f_y(x_i, y_{io})$$

and $$S_i'(y_{in}) = f_y(x_i, y_{in})$$ if they are known

or approximate derivatives found by differentiating a cubic Lagrange polynomial if they are not. (As in 3.2) .

We will then find an approximation to the area under the curve $f(x_i,y)$ along the line $x=x_i$ which is

$$\int_{y_{io}}^{y_{in}} f(x_i,y)\ dy$$

by evaluating the integral

$$\int_{y_{io}}^{y_{in}} S_i(y)\ dy \quad .$$

Let these approximate areas be denoted by $Y(x_i)$.

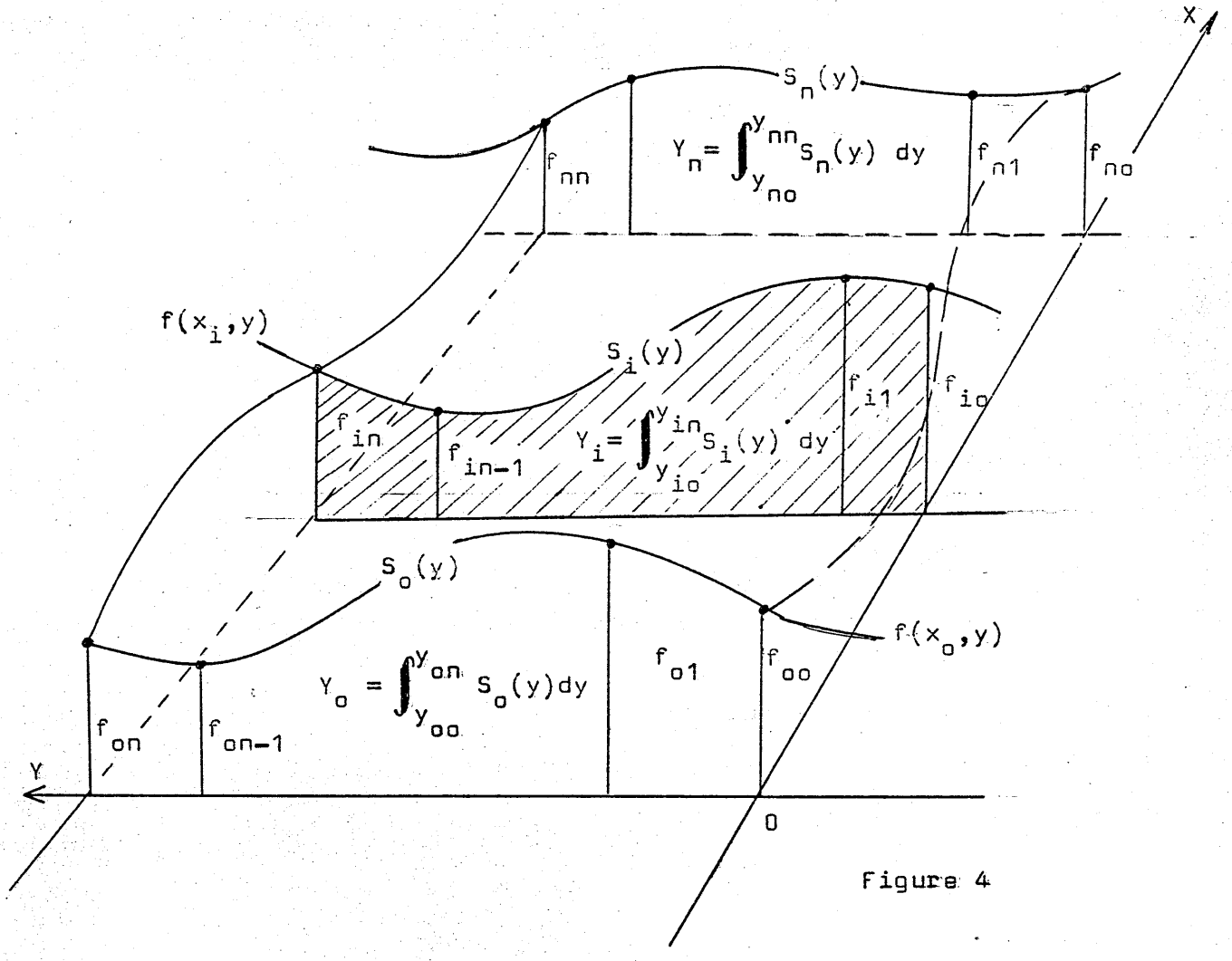


Figure 4

We will next fit a one dimensional cubic spline approximation $S_v(x)$ to the values $Y(x_i)$ $i=0,\ldots,n$. Since the $Y(x_i)$ must be treated as data points whose parent function is not known, then we must compute boundary conditions also. For this we use the cubic Lagrange polynomial and differentiate to find approximations for f_o' and f_n'. We thus find the approximate value of the double integral

$$\int_R f(x,y)\ dxdy$$

from the integral $\int_{x_o}^{x_n} S_v(x)\,dx$.

4.3 Fitting the cubic spline of section 3.

As an alternative to the method described in 4.2 for evaluating an approximation to the doble integral we can apply the cubic spline approximation of section 3.

Along the line $x=x_i$ we define the function $F_i(y)$ as

$$F_i(y) = \int_{y_{io}}^{y} f(x_i,t)\,dt$$

and fit the cubic spline $S_i(y)$ to the function $F_i(y)$ at the points $y=y_{io}$, ... , $y=y_{in}$ as we did in section 3 so that the integral

$$\int_{y_{io}}^{y_{in}} f(x_i,y)\,dy$$ is approximated by the value

$$S_i(y_{in}) .$$

If these integrals are denoted by $Y(x_i)$, $i=0,...,n$, as in 4.2 then we define $F_v(x)$ as

$$F_v(x) = \int_{x_o}^{x} Y(t)\,dt$$

and fit a cubic spline $S_v(x)$ to the values $F_v(x_i)$, again as we did in section 3, and our approximation to the double integral

$$\int_R f(x,y)\,dxdy$$

is $S_v(x_n)$.

4.4 Numerical Results

Given below is a table of results found from using the two methods described above. The method of 4.2 is (1) and the method of 4.3 is (2) .

These results were computed on a TRS 80 microcomputer which, in single precision arithmetic, gives reults in six significant figures.

It is noticeable that method (2) is more accurate for these particular integrals.

Function	Interval	mesh size	Numerical results	True values	Absolute errors
$\sin x \sin y$	$0 \leqslant x \leqslant \frac{1}{2}\pi$ $0 \leqslant y \leqslant \frac{1}{2}\pi$	$h_x = \pi/20$ $h_y = \pi/20$	(1) 1.000030 (2) 1.000010	1.000000	0.000030 0.000010
$\cos x \cos y$	$0 \leqslant x \leqslant \frac{1}{2}\pi$ $0 \leqslant y \leqslant \frac{1}{2}\pi$	$h_x = \pi/20$ $h_y = \pi/20$	(1) 1.000030 (2) 1.000010	1.000000	0.000030 0.000010
$\frac{4}{\pi}\exp(-(x^2+y^2))$	$0 \leqslant x \leqslant 1$ $0 \leqslant y \leqslant 1$	$h_x = 0.1$ $h_y = 0.1$	(1) 0.710151 (2) 0.710147	0.710145	0.000006 0.000002
$1/xy$	$1 \leqslant x \leqslant 2$ $1 \leqslant y \leqslant 2$	$h_x = 0.1$ $h_y = 0.1$	(1) 0.480468 (2) 0.480457	0.480453	0.000015 0.000002

4.5 Computer programme for (1)

```
10 INPUT N:DIMX(N),Y(N),F2(N,N),A(N,2),M(N),B(N),C(N),S(N),F(N)
20 INPUT X(0),X(N),Y(0),Y(N):HX=(X(N)-X(0))/N:HY=(Y(N)-Y(0))/N
30 FORI=0TON:X(I)=X(0)+I*HX:Y(I)=Y(0)+I*HY:NEXTI:FORI=0TON
40 FORJ=0TON:XI=X(I):YJ=Y(J):GOSUB1000:F2(I,J)=FXY:NEXTJ,I
```

This has set up the mesh points (x_i, y_j) and defined the function values $f(x_i, y_j)$.

```
50 H=HY:FORI=0TON:FORJ=0TON:F(J)=F2(I,J):C(J)=Y(J):NEXTJ:GOSUB500
```

Subroutine 500 computes the moments M_i of the linear cubic spline along the line $x=x_i$

```
FORJ=1TON:S(I)=S(I)+(F(J)+F(J-1))*H/2-(M(J)+M(J-1))*(H↑3)/24
70 NEXTJ,I
```

The vector S holds the values of the areas $\int_{y_o}^{y_n} f(x_i, y)\, dy$.

```
80 H=HX:FOR I=0TON:F(I)=S(I):C(I)=X(I):NEXTI: GOSUB500
```

We have computed the moments of a linear cubic spline fitted to the values in S at the points where $x=x_i$.

```
90 FORI=1TON:
100 SUM=SUM+(F(I)+F(I-1))*H/2-(M(I)+M(I-1))*(H↑3)/24:NEXTI
110 PRINT SUM:END

500A(0,1)=H/3:A(0,2)=H/6:A(N,0)=H/6:A(N,1)=H/3:XI=C(0):K=0
510 GOSUB1100:B(0)=(F(1)-F(0))/H-FD:K=1:XI=C(N):GOSUB1100
B(N)=FD-(F(N)-F(N-1))/H:FORIA=1TON-1:A(IA,0)=H/6:A(IA,1)=(2*H)/3
530 A(IA,2)=H/6:B(IA)=(F(IA+1)-2*F(IA)+F(IA-1))/H:NEXTIA
540 FORIA=1TON:A(IA,1)=A(IA,1)- A(IA-1,2)*A(IA,0)/A(IA-1,1)
550 B(IA)=B(IA)-B(IA-1)*A(IA,0)/A(IA-1,1):A(IA,0)=0:NEXTIA
560 M(N)=B(N)/A(N,1):FORIA=N-1TOOSTEP-1
570 M(IA)=(B(IA)-A(IA,2)*M(IA+1))/A(IA,1):NEXTIA:RETURN

1000 FXY= ......    :RETURN

1100 IF K=0 THEN GOTO 1110 ELSE GOTO 1120
1110 FD=-(3*F(0))/(2*H)+(2*F(1))/H-F(2)/(2*H):GOTO 1130
1120 FD=F(N-2)/(2*H)-(2*F(N-1))/H+(3*F(N))/(2*H)
1130 RETURN
```

4.6 Computer Programme for (2)

```
10 INPUT N:DIM S(N),F(N),F2(N,N),X(N),Y(N),M(N),S2(N),C(N)
20 INPUT X(0),X(N),Y(0),Y(N):HX=(X(N)-X(0))/N:HY=(Y(N)-Y(0))/N
30 FORI=0TON:X(I)=X(0)+I*HX:Y(I)=Y(0)+I*HY:NEXTI
40 FORI=0TON:FORJ=0TON:XI=X(I):YJ=Y(J):GOSUB1000:F2(I,J)=FXY
50 NEXT J,I:H=HY
60 FORI=0TON:FORJ=0TON:F(J)=F2(I,J):NEXTJ:GOSUB1100:M(0)=FD
70 GOSUB500:S2(I)=S(N):NEXTI:H=HX
```

The vector S2 contains the areas $\int_{y_0}^{y_n} f(x_i,y)dy$ for $i=0,..,n$.

```
80 FORI=0TON:F(I)=S2(I):NEXTI:GOSUB1100:M(0)=FD:GOSUB500
90 PRINT S(N):END
```

Subroutine 500 computes areas using the cubic spline of section 3

```
500 FORK=0TON-1:M(K+1)=2*(F(K+1)-F(K))/H-M(K):NEXTK
510 S(0)=0:FORK=1TON
520 C(K)=S(K-1)-M(K-1)*(H*X(K-1)+(H 2)/3)/2-F(K-1)*X(K-1)
530 S(K)=M(K)*(H 2)/6+M(K-1)*H*X(K)/2+F(K-1)*X(K)+C(K)
540 NEXTK:RETURN.
```

C is the vector which holds the constants of integration C_{2i} and S(N) is the area.

```
1000 FXY= ....    :RETURN        Function subroutine

1100 FD=-(3*F(0))/(2*H)+(2*F(1))/H-F(2)/(2*H):RETURN
```

4.7 These methods of section 4 follow the steps in the analytical proceedure for integrating a function of two variables. We require to evaluate

$$V = \int_R f(x,y)\ dxdy$$

where R is a region of the x-y plane bounded by closed curve C. If f(x,y) and R were such that an analytical proceedure for integrating were applicable then we would first integrate f(x,y) with respect to one variable, say y, and thereby determine a continuous function of x, I(x), which would be defined as

$$I(x) = \int_y f(x,t)\ dt \quad .$$

In particular, along the line $x=x_i$ in the x-y plane, parallel to the y axis, we would have

$$I(x_i) = \int_{y_{io}}^{y_{in}} f(x,y)\ dy$$

where y_{in} and y_{io} $(y_{in} \leqslant y_{io})$ are the values of y where the line $x=x_i$ intersects the curve C. For simplicity, we assume that any line $x=x_i$ intersects C in at most two points as in 4.1.
We would now determine V from

$$V = \int_{x_o}^{x_n} I(x)\ dx$$

where x_o and x_n $(x_n \leqslant x_o)$ are the extreme values of x on C .
In following this proceedure with our algorithms of 4.2 and 4.3, however., we are only in a position to find approximate values. We cannot determine $I(x_i)$ exactly but only an approximation, $I_p(x_i)$, which is found by fitting a one dimensional cubic spline along the line $x=x_i$ to the ordinate values $f(x_i,y_j)$ $j=0,..,n$. If we had the true values of $I(x_i)$ at $x=x_i$ for $i=0,..,n$, then we could fit a one dimensional cubic spline $S_I(x)$ to these values of I(x) and then integrate this cubic spline to find an approximate value for V. However, since we know only $I_p(x_i)$, $i=0,..,n$, in fact, we therefore fit the one dimensional cubic spline $S_p(x)$ to these values and approximate V by

$$\int_{x_o}^{x_n} S_p(x)\ dx \quad .$$

The total error in V by so doing is

$$\int_{x_o}^{x_n}(I - S_p)\,dx \quad .$$

Since we could, in theory, find an $I_p(x)$ for any value of x, where $x_o \leqslant x \leqslant x_n$, as an approximation to I(x), then $I_p(x)$ can also be considered as a continuous function of x.
$I_p(x)$ then, is the approximation to I(x) found by fitting a cubic spline along any line in the x-y plane, parallel to the y axis, between the points where the line intersects the boundary curve C and $x_o \leqslant x \leqslant x_n$, then integrating this cubic spline with respect to y.

Now $$I - S_p = I - I_p + I_p - S_p$$

so that $$\int_{x_o}^{x_n}(I-S_p)\,dx = \int_{x_o}^{x_n}(I-I_p)\,dx + \int_{x_o}^{x_n}(I_p-S_p)\,dx$$

and $$\left|\int_{x_o}^{x_n}(I - S_p)\,dx\right| \leqslant \left|\int_{x_o}^{x_n}(I - I_p)\,dx\right| + \left|\int_{x_o}^{x_n}(I_p - S_p)\,dx\right|$$

A bound for $|I - I_p|$ can be found from 2.5.7 since I_p is the approximation to I found by integrating a cubic spline.

$$\left|I(x_i) - I_p(x_i)\right| \leqslant \frac{5}{384} h_y^4 (y_{in} - y_{io}) \left\|f^{(0,4)}\right\|$$

where, along the line $x=x_i$, h_y is the maximum distance between any two consecutive mesh points and $\left\|f^{(0,4)}\right\|$ is the maximum absolute value of the fourth partial derivative with respect to y. A bound for

$$\left|\int_{x_o}^{x_n}(I - I_p)\,dx\right|$$

is therefore

$$\frac{5}{384} h_y^4 (y_n - y_o)(x_n - x_o) \left\|f^{(0,4)}\right\| \quad \ldots\ldots 4.7.1$$

where, on the boundary curve C, y_o and y_n, x_o and x_n, are the extreme values of y and x, respectively, h_y is the maximum distance between any two consecutive mesh points along any line parallel to the y axis and $\left\|f^{(0,4)}\right\|$ is the maximum absolute value of the fourth partial derivative with respect to y anywhere in the region R.

Since S_p is a one dimensional cubic spline approximation to I_p then $|I_p - S_p|$ is bounded by

$$\frac{5}{384} h_x^4 \left\| I_p^{iv}(x) \right\|$$

where h_x is the maximum distance between any two consecutive mesh points on the mesh $x_o \leqslant x_1 \ldots \leqslant x_n$ and $I^{iv}(x)$ is the maximum absolute value of the fourth derivative with respect to x in $[x_o, x_n]$. A bound for the integral

$$\left| \int_{x_o}^{x_n} (I_p - S_p)\, dx \right| \quad \text{is therefore}$$

$$\frac{5}{384} h_x^4 (x_n - x_o) \left\| I_p^{iv}(x) \right\| . \qquad \ldots\ldots 4.7.2$$

A numerical value for this error bound hinges on an estimate for $\left\| I_p^{iv}(x) \right\|$. This is difficult. We have only intimated the existence of $I_p(x)$ as a continuous function by intuition but we could show that $I_p(x)$ is one of a sequence of functions which will converge to $I(x)$. In practice, we would only like to choose an approximation which does converge. Certainly, in this case, and in the examples of 4.4, such convergence is implicitly believed. We would therefore be led to believe that we can approximate $I_p^{iv}(x)$ using $I^{iv}(x)$ without danger provided that we <u>do</u> understand how and why we have done so.

By definition

$$I(x_i) = \int_{y_{io}}^{y_{in}} f(x_i, y)\, dy$$

along the line $x=x_i$ so that, along any line parallel to the y axis which cuts C in points where $y=y_o(x)$ and $y=y_n(x)$ we have

$$I(x) = \int_{y_o(x)}^{y_n(x)} f(x,y)\, dy \quad .$$

Differentiating under the integral sign:

$$I^{iv}(x) = \int_{y_o(x)}^{y_n(x)} f^{(4,0)}(x,y)\, dy$$

By the Mean Value Theorem for integrals, we can find η such that $y_o(x) \leqslant \eta \leqslant y_n(x)$ and

$$\int_{y_o(x)}^{y_n(x)} f^{(4,0)}(x,y)\,dy = (y_n(x)-y_o(x))\, f^{(4,0)}(x,\eta)\ .$$

If $\|f^{(4,0)}\|$ is the maximum value of the fourth partial derivative of $f(x,y)$ in the region R and on the closed curve C, then

$$\|I^{iv}(x)\| \leqslant (y_n - y_o)\ \|f^{(4,0)}\|$$

and so

$$\left| \int_{x_o}^{x_n} (I_p - S_p)\,dx \right| \leqslant \frac{5}{384} h_x^4\, (x_n - x_o)(y_n - y_o)\ \|f^{(4,0)}\| \qquad \ldots\ldots\ 4.7.3$$

Combining 4.7.1 and 4.7.3 we find

$$\left| \int_{x_o}^{x_n} (I - S_p)\,dx \right| \leqslant \frac{5}{384}(y_n - y_o)(x_n - x_o)\ \left(h_x^4\ \|f^{(4,0)}\| + h_y^4\ \|f^{(0,4)}\| \right) \qquad \ldots\ldots\ 4.7.4$$

It will have been noticed that, in the foregoing analysis, we have used the less optimistic error bound 2.5.7 instead of the bound given by 2.5.5. This is done to allow for the assumption that $I^{iv}(x)$ could be used in the place of $I_p^{iv}(x)$ since the latter is not known. If we wer to be very optimistic about the substitution of $I^{iv}(x)$ for $I_p^{iv}(x)$ in the error analysis, then we could say that

$$\left| \int_{x_o}^{x_n} (I - S_p)\,dx \right| \leqslant \frac{1}{720} (y_n - y_o)(x_n - x_o)\left(h_x^4\ \|f^{(4,0)}\| + h_y^4\ \|f^{(0,4)}\| \right) \qquad \ldots\ldots\ 4.7.5$$

5. COMPUTATION OF DOUBLE INTEGRALS USING THE BICUBIC SPLINE

5.1 The Bicubic Spline

Let $S(x,y)$ be the bicubic spline approximation to the function $f(x,y)$ of two variables x and y over the region R of the x-y plane: $x_o \leqslant x \leqslant x_n$, $y_o \leqslant y \leqslant y_n$ where R is subdivided into a rectangular mesh by the lines $x=x_i$ $i=1,\ldots,n-1$ and $y=y_j$ $j=1,\ldots,n-1$.

By analogy with the one dimensional case, the function $S(x,y)$ is continuous with continuous first partial and second mixed partial derivatives, is a bicubic polynomial in each mesh rectangle r_{ij} : $x_{i-1} \leqslant x \leqslant x_i$, $y_{j-1} \leqslant y \leqslant y_j$ for $i=1,..,n$ $j=1,..,n$, and solves the interpolation problem

$$\begin{aligned} S(x_i,y_j) &= f(x_i,y_j) && i=0,..,n \quad j=0,..,n \\ S_x(x_i,y_j) &= f_x(x_i,y_j) && i=0,n \quad j=0,..,n \\ S_y(x_i,y_j) &= f_y(x_i,y_j) && i=0,..,n \quad j=0,n \\ S_{xy}(x_i,y_j) &= f_{xy}(x_i,y_j) && i=0,n \quad j=0,n \end{aligned} \qquad \ldots\ldots 5.1.1$$

In each mesh rectangle r_{ij} $S(x,y)$ is defined by the bicubic polynomial

$$s_{ij}(x,y) = \sum_{u=0}^{3} \sum_{v=0}^{3} (c_{ij})_{uv}\, x^u y^v \qquad \ldots\ldots 5.1.2$$

It is proved by De Boor[6] that $S(x,y)$ of the form 5.1.2 in each mesh rectangle and satisfying the constraints 5.1.1 over R is unique.

The Bicubic Hermite polynomial which fits the values of $f(x,y)$, $f_x(x,y)$, $f_y(x,y)$ and $f_{xy}(x,y)$ at all the mesh points (x_i,y_j) for $i=0,..,n$ and $j=0,..,n$, is also a Bicubic Spline but the reverse is not always true. The Bicubic Spline $S(x,y)$ does not necessarily fit the values of the the partial derivatives of $f(x,y)$ but the partial derivatives of $S(x,y)$ must be continuous at the internal mesh points.

The 16 coefficients $(c_{ij})_{uv}$ in the ijth rectangle r_{ij} can be determined from the 16 linear equations

$$\begin{aligned} s_{ij} &= f_{ij} \\ (s_{ij})_x &= f_x(x_i,y_j) \\ (s_{ij})_y &= f_y(x_i,y_j) \end{aligned}$$

and $(s_{ij})_{xy} = f_{xy}(x_i,y_j)$

which are obtained at the four corners of r_{ij} :

(x_i,y_j) , (x_i,y_{j-1}) , (x_{i-1},y_j) and (x_{i-1},y_{j-1}) , but we are then taxed with the problem of finding approximate values for the necessary partial derivatives of $f(x,y)$ at the mesh points. Under the terms of the constraints 5.1.1 the values of $f(x,y)$ at the mesh points are expected to be given, together with the values of f_x along the lines $y=y_o$, $y=y_n$, the values of f_y along the lines $x=x_o$, $x=x_n$ and the values of f_{xy} at the four corners of R. In practice, however, the application of the bicubic spline to the problem of interpolation would probably involve fitting $S(x,y)$ to a set of data points $f(x_i,y_j)$ $i=0,..,n$, $j=0,..,n$ without any derivative values being given. We assume here that these values <u>will</u> be given.

We compute the approximate values of f_x, f_y and f_{xy} at the internal mesh points using one dimensional cubic splines along the lines $x=x_i$ and $y=y_j$, for $i=1,..,n-1$ and $j=1,..,n-1$. In particular, to compute f_x along each line $y=y_j$, we fit the cubic spline $s_j(x,y_j)$ to the function values $f(x_i,y_j)$ $i=0,..,n$ along the line and the boundary derivatives $f_x(x_o,y_j)$, $f_x(x_n,y_j)$. From this cubic spline we find, by differentiating $s_j(x,y_j)$ with respect to x, the approximate values of $f_x(x_i,y_j)$ $i=1,..,n-1$, denoted by p_{ij} ,and build up a table of these values over R . At the boundary of R $p_{oj}= f_x(x_o,y_j)$ and $p_{nj}= f_x(x_n,y_j)$. Similarly, along each line $x=x_i$ we fit a one dimensional cubic spline $s_i(x_i,y)$ to the function values $f(x_i,y_j)$ for $j=0,..,n$ at the mesh points along this line and the boundary derivatives $f_y(x_i,y_o)$, $f_y(x_i,y_n)$. From this one dimensional cubic spline we compute approximate values of $f_y(x_i,y_j)$, denoted by q_{ij}, by differentiating $s_i(x_i,y)$ with respect to y. We build up a table of values of q_{ij} over R. At the boundary of R, the values of $q_{io}= f_y(x_i,y_o)$ and $q_{in}= f_y(x_i,y_n)$ are given .

At the mesh points we can compute approximate values for $f_{xy}(x_i,y_j)$, denoted by t_{ij}. First, along the lines $y=y_o$ and $y=y_n$ at the boundary, we fit one dimensional cubic splines $s_{q_o}(x,y_o)$ and $s_{q_n}(x,y_n)$ to the known values of f_y with the known values of f_{xy} at the relevant corners of R. By differentiating these splines with respect to x we find values for t_{io} and t_{in} for $i=1,..,n-1$.

Along each line $x=x_i$ $i=0,..,n$ we then fit the one dimensional cubic spline $s_{p_i}(x_i,y)$ to the values of p_{ij} at the mesh points along the line and the values of t_{io} and t_{in} at the ends of the line. By differentiating $s_{p_i}(x_i,y)$ with respect to y we find values for t_{ij} $j=1,..,n-1$ at the mesh points.
We now have three tables of values over the region R : p_{ij}, q_{ij} and t_{ij}, each with n^2 entries, and we have the n^2 given values of f_{ij} in R . In each mesh rectangle r_{ij} we therefore have the necessary 16 values to form the required set of 16 linear equations from which to compute the 16 coefficients $(c_{ij})_{uv}$ $(u=0,..,3\ v=0,..,3)$ of the bicubic spline 5.1.1 .

5.2 The integral of the Bicubic Spline.

Having found the values of the $(c_{ij})_{uv}$ then the approximate value for the integral

$$\int_R f(x,y)\,dxdy$$

is

$$\sum_{i=o}^{n}\sum_{j=o}^{n}\ \sum_{u=o}^{3}\sum_{v=o}^{3}(c_{ij})_{uv}\ \frac{x^{u+1}}{u+1}\ \frac{y^{v+1}}{v+1} \qquad \ldots\ldots\ 5.2.1$$

5.3 The Computer Programme

The computer programme is written to evaluate the integral

$$\int_{x_o}^{x_n}\int_{y_o}^{y_n} f(x,y)\,dxdy$$

on a rectangular mesh $x_o \leqslant x \leqslant x_n$, $y_o \leqslant y \leqslant y_n$. The programme is in BASIC .

```
10 INPUT N:DIM F(N,N),P(N,N),Q(N,N),R(N,N),A(N,2),C(15,15)
20 DIM M(N),FS(N),B(N),X(N),Y(N),FD(1),XM(1),YM(1),CA(15)
30 DIM CB(16),CUV(15)
40 INPUT X(0),X(N),Y(0),Y(N):HX=(X(N)-X(0))/N:HY=(Y(N)-Y(0))/N
50 FORIA=0TON:X(IA)=X(0)+IA*HX:Y(IA)=Y(0)+IA*HY:NEXTIA
```

The mesh points (x_i,y_j) have been set up.

```
60 FORIA=0TON:FORJA=0TON:XI=X(IA):YI=Y(JA):GOSUB1100:FS(JA)=F
70 F(IA,JA)=F:NEXTJA .....
```

This sets up the array of values $f(x_i,y_j)$ and a set of n+1 function (FS) values along each line $x=x_i$.

```
..... H=HY:YJ=Y(0):GOSUB1200:FD(0)=FY:YJ=Y(N)
80 GOSUB1200:FD(1)=FY....
```

This gives us boundary values along the line $x=x_i$.

```
..... GOSUB1000:Q(IA,0)=FD(0):Q(IA,N)=FD(1)
90 FORJA=1TON-1:Q(IA,JA)=(2*M(JA)+M(JA-1))*H/6+(FS(JA)-FS(JA-1))/H
100 NEXTJA:NEXTIA
```

Subroutine 1000 computes the moments M_i and Q is the array of derivatives $f_y(x_i,y_j)$.

```
110 FORJA=0TON:YJ=Y(JA):FORIA=0TON:FS(IA)=F(IA,JA):NEXTIA
120 H=HX:XI=X(0):GOSUB1150:FD(0)=FX:XI=X(N):GOSUB1150:FD(1)=FX
130 GOSUB1000:P(0,JA)=FD(0):P(N,JA)=FD(1):FORIA=1TON-1
140 P(IA,JA)=(2*M(IA)+M(IA-1))*H/6+(FS(IA)-FS(IA-1))/H
150 NEXTIA:NEXTJA
```

P is the array of derivatives $f_x(x_i,y_j)$.

```
160 FORJA=0TON:YJ=Y(JA):FORIA=0TON:FS(IA)=Q(IA,JA):NEXTIA
170 H=HX:XI=X(0):GOSUB1250:FD(0)=F2XY:XI=X(N):GOSUB1250:FD(1)=F2XY
180 GOSUB1000:R(0,JA)=FD(0):R(N,JA)=FD(1):FORIA=1TON-1
190 R(IA,JA)=(2*M(IA)+M(IA-1))*H/6+(FS(IA)-FS(IA-1))/H
200 NEXTIA:NEXTJA
```

R is the array of second mixed partial derivatives $f_{xy}(x_i,y_j)$.

In each mesh rectangle $x_{i-1} \leqslant x \leqslant x_i$, $y_{j-1} \leqslant y \leqslant y_j$ we now set up the system of 16 linear equations from which to compute the coefficients C_{uv} of the bicubic polynomial elememt in that rectangle.

```
210 FORIA=1TON:FORJA=1TON
220 XM(0)=X(IA):XM(1)=X(IA-1):YM(0)=Y(JA):YM(1)=Y(JA-1)
230 FORIB=0TO1:FORJB=0TO1:FORU=1TO4:FORV=1TO4
240 K=U*V+(U-1)*(4-V)-1:KA=IB*2+JB
250 C(KA,K)=(XM(IB)↑(U-1))*(YM(JB)↑(V-1)):NEXTV,U
260 CA(KA)=F(IA-IB,JA-JB):NEXTJB,IB
270 FORIB=0TO1:FORJB=0TO1:FORU=2TO4:FORV=1TO4
280 K=U*V+(U-1)*(4-V):KA=IB*2+JB
290 C(KA+4,K)=(U-1)*(XM(IB)↑(U-2)*(YM(JB)↑(V-1)):NEXTV,U
300 CA(KA+4)=P(IA-IB,JA-JB):NEXTJB,IB
310 FORIB=0TO1:FORJB=0TO1:FORU=1TO4:FORV=2TO4
320 K=U*V+(U-1)*(4-V)-1:KA=IB*2+JB
330 C(KA+8,K)=(XM(IB)↑(U-1))*(V-1)*(YM(JB)↑(V-2)):NEXTV,U
340 CA(KA+8)=Q(IA-IB,JA-JB):NEXTJB,IB
350 FORIB=0TO1:FORJB=0TO1:FORU=2TO4:FORV=2TO4
360 K=U*V+(U-1)*(4-V)-1:KA=IB*2+JB
370 C(KA+12,K)=(U-1)*(XM(IB)↑(U-2))*(V-2)*(YM(JB)↑(V-2)):NEXTV,U
380 CA(KA+12)=R(IA-IB,JA-JB):NEXTJB,IB
```

Next we solve this system of 16 linear equations using Gauss elimination with pivoting:

```
390 FORK=0TO14:KA=K+1:FORKB=KATO15
400 IFABS(C(KB,K) ABS(C(K,K)THENGOTO430ELSEGOTO410
410 FORL=0TO15:CB(L)=C(KB,L):C(KB,L)=C(K,L):C(K,L)=CB(L):
420 NEXTL:CB(16)=CA(KB):CA(KB)=CA(K):CA(K)=CB(16)
430 NEXTKB
440 FORKB=KATO15:CM=C(KB,K)/C(K,K)            CM is the multiplier
450 FORL=KTO15:C(KB,L)=C(KB,L)-CM*C(K,L):NEXTL
460 CA(KB)=CA(KB)-CM*CA(K):NEXTKB:NEXTK
```

At this stage the matrix C is triangular

```
470 CUV(15)=CA(15)/C(15,15)
480 FORK=14TOOSTEP-1:KA=K+1:FORKB=KATO15
CA(K)=CA(K)-C(K,KB)*CUV(KB):NEXTKB
500 CUV(K)=CA(K)/C(K,K):NEXTK
```

Now that we have computed the coefficients of the bicubic spline elememt we can integrate in that rectangle.

```
505 FORI=0TON:FORJ=0TON:C(I,J)=0:NEXTJ,I
510 FORU=1TO4:FORV=1TO4:K=U*V+(U-1)*(4-V)
520 S1=CUV(K)*(X(IA) U-X(IA-1) U)*(Y(JA) V-Y(JA-1) V)/(U*V)
530 SUM=SUM+S1:NEXTV,U:NEXTJA,IA:PRINTSUM:END
```

Subroutine 1000 computes the moments M_i of the cubic spline.

```
1000 A(0,1)=H/3:A(0,2)=H/6:A(N,0)=H/6:A(N,1)=H/3
1010 B(0)=(FS(1)-FS(0))/H-FD(0):B(N)=FD(1)-(FS(N)-FS(N-1))/H
1020 FORID=1TON-1:A(ID,0)=H/6:A(ID,1)=(2*H)/3:A(ID,2)=H/6
1030 B(ID)=(FS(ID+1)-2*FS(ID)+FS(ID-1))/H:NEXTID
1040 FORID=1TON:A(ID,1)=A(ID,1)-A(ID,2)*A(ID,0)/A(ID-1,1)
1050 B(ID)=B(ID)-B(ID-1)*A(ID,0)/A(ID-1,1):A(ID,0)=0:NEXTID
1060 M(N)=B(N)/A(N,1):FORID=N-1TO0STEP-1
1070 M(ID)=(B(ID)-A(ID,2)*M(ID+1))/A(ID,1):NEXTID:RETURN
```

```
1100 F= .......    :RETURN    Function subroutine
1150 FX= .......   :RETURN    f_x subroutine
1200 FY= .......   :RETURN    f_y subroutine
1250 F2XY= ....    :RETURN    f_xy subroutine
```

5.4 Numerical results

The computer programme as laid out in section 5.3 takes an exceedingly long time to execute. When the region is divided up into only 4 mesh rectangles (n=2) it takes approximately 10 minutes to compute the integral over the region. Admittedly, not much thought went into the economy of programming and it obviously would help to review the programme with this in mind, but, nevertheless it is a long execution for results which can be obtained by the less complicated methods of sections 4. We may find a considerable saving in computation time, without loss, in using difference methods to compute the required derivatives at the mesh points.

The programme was first tested using the function $f(x,y)=x^3y^3$. It gave the exact result.

The following results were obtained with other functions:

Function	Region	n	Integral	Exact Result
sin x sin y	$0 \leq x \leq \frac{1}{2}\pi$ $0 \leq y \leq \frac{1}{2}\pi$	2	0.996294	1,000000
1/xy	$1 \leq x \leq 2$ $1 \leq y \leq 2$	2	0.479504	0.480450

5.5 Error Bounds for Integral Approximation using Bicubic Splines

Consider the region R of the x-y plane

$$R: x_o \leqslant x \leqslant x_n \ , \ y_o \leqslant y \leqslant y_n$$

over which a function f of the two variables x and y are defined and which belongs the the space $\mathcal{C}_2^4(R)$. R is divided up into a rectangular mesh by lines $x=x_i$, $y=y_j$ $(i=1,..,n-1 \quad j=1,..,n-1)$ and f is approximated to by a bicubic spline.

If, at the corners of the mesh rectangle r_{ij}, in addition to the values of f, the values of $f^{(1,0)}$, $f^{(0,1)}$ and $f^{(1,1)}$ can be accurately determined, then we can fit the bicubic spline Z(x,y) to f over R where, in r_{ij}

$$Z(x,y) = z_{ij}(x,y) = \sum_{u=0}^{3} \sum_{v=0}^{3} (b_{ij})_{uv} \, x^u y^v$$

and an error bound in approximating f using $z_{ij}(x,y)$ in r_{ij} is given by

$$\|f - z_{ij}\|$$

$$\leqslant \frac{5}{384} \|f^{(4,0)}\| h_x^4 + \frac{81}{64} \|f^{(2,2)}\| h_x^2 h_y^2 + \frac{5}{384} \|f^{(0,4)}\| h_y^4 \quad .. 5.5.1$$

where h_x and h_y are the lengths of the sides of r_{ij} and the norms are the maximum values in r_{ij} . (Carlson & Hall[10]) .

The method of section 5 does, however, introduce additional errors because the bicubic spline coefficients are computed using approximate values for the partial derivatives of the function f. On a regular mesh the error bound for the first partial derivatives computed using one dimensional cubic splines, is of order h^3, where $h=h_x$ or $h=h_y$ as appropriate, and the error bound for the second mixed partial derivative is of order h^2 where $h=\max(h_x,h_y)$.

We have $$\|f^{(1,0)} - p'_j\| \leqslant k_1 h_x^3 \quad \ldots\ldots \quad 5.5.2 \text{ (a)}$$

$$\|f^{(0,1)} - q'_i\| \leqslant k_1 h_y^3 \quad \ldots\ldots \quad 5.5.2 \text{ (b)}$$

$$\|f^{(1,1)} - t'\| \leqslant k_2 h^2 \quad \ldots\ldots \quad 5.5.3$$

p'_j , q'_i and t' are the approximate values of the partial derivatives as computed in section 5.1 and k_1 and k_2 as adapted for the regular mesh from those given by C.A,Hall(7) , are

$$k_1 = \left(\frac{\sqrt{3}}{216} + \frac{1}{24} \right) \|f^{iv}\| \qquad k_2 = \frac{5}{12} \|f^{iv}\|$$

where, here, we take $\|f^{iv}\| = \max(\|f^{(m,n)}\| \ , \ m+n=4)$.

To compute the 16 coefficients $(c_{ij})_{uv}$, of $s_{ij}(x,y)$ in r_{ij} as we do in 5.1 where

$$s_{ij}(x,y) = \sum_{u=0}^{3} \sum_{v=0}^{3} (c_{ij})_{uv} \, x^u y^v$$

we find the system of linear equations

$$AC=F$$

The matrix A contains the bicubic polynomial terms $x^u y^v$ (u=0,.,3 v=0,.,3), without the coefficients $(c_{ij})_{uv}$, of $s_{ij}(x,y)$ and its first and second mixed partial derivatives at the corners of r_{ij}. The vector C contains the coefficients $(c_{ij})_{uv}$ which are to be determined. The vector F contains the values of f, p'_{ij}, q'_{ij} and t'_{ij} at the corners of r_{ij} .
Twelve of the elements of F are the approximate values of the partial derivatives of f. If the true values of the function f and its partial derivatives at the corners of r_{ij} are held in the vector F' and the vector δF contains the error bounds for F then

$$\|F - F'\| \leqslant \|\delta F\|$$

taking the maxima over r_{ij} . $\|\delta F\|$ is therefore equal either to the appropriate part of 5.5.2 or to 5.5.3 whichever is the greater.
If we take the m-norm of the matrix A^{-1}, where the m-norm of the matrix A is defined as

$$\|A\|_m = \max_p \left(\sum_q |a_{pq}| \right) ,$$

then an error bound for any element of the vector C (a bound for the value $|(b_{ij})_{uv} - (c_{ij})_{uv}|$ for any u=0,.,3 v=0,.,3) is

$$\|A^{-1}\|_m \ \|\delta F\| \qquad \ldots\ldots \ 5.5.4$$

It should be understood that this error bound is a bound for the absolute value of the difference between any two corresponding members of two sets of coefficients - a theoretical set $(b_{ij})_{uv}$ which would be found if the true values of f and its partial derivatives at the corners of r_{ij} were known and those computed

by means of the algorithm in 5.1. The error bound 5.5.1 is stated in the belief that the theoretical set of coefficients $(b_{ij})_{uv}$ are accurately known. To the error bound 5.5.1 therefore, we must add another term.

We have

$$\left|z_{ij}(x,y) - s_{ij}(x,y)\right| \leqslant \sum_{u=0}^{3}\sum_{v=0}^{3} \left|(b_{ij})_{uv} - (c_{ij})_{uv}\right| \left|x^u y^v\right|$$

Let T_{uv} be the maximum absolute value of any product $x^u y^v$ $(u=0,.,3$ $v=0,.,3)$ in r_{ij} then

$$\left\|z_{ij} - s_{ij}\right\| \leqslant 16\, T_{uv} \left\|A^{-1}\right\|_m \left\|\delta F\right\| \quad .$$

Since $\quad f - s_{ij} = (f - z_{ij}) + (z_{ij} - s_{ij})$

then $\quad \left\|f - s_{ij}\right\| \leqslant \left\|f - z_{ij}\right\| + \left\|z_{ij} - s_{ij}\right\|$

so that, in r_{ij} :

$$\left\|f - s_{ij}\right\| \leqslant$$

$$\frac{5}{384}\left\|f^{(4,0)}\right\| h_x^4 + \frac{81}{64}\left\|f^{(2,2)}\right\| h_x^2 h_y^2 + \frac{5}{384}\left\|f^{(0,4)}\right\| h_y^4$$

$$+ 16 T_{uv} \left\|A^{-1}\right\|_m \left\|\delta F\right\| \quad \ldots \quad 5.5.5$$

If T_{uv} , $\left\|A^{-1}\right\|_m$, $\left\|\delta F\right\|$ and the norms for the partial derivatives are the maximum that occur anywhere in the region R, the we find a bound for $\left\|f - S\right\|$ in R is 5.5.5 also.

A bound then for the absolute value of the integral

$$\int_R (f - S)\, dxdy \quad \text{is}$$

$$\left|\int_R (f - S)\, dxdy\right|$$

$$\leqslant (x_n - x_o)(y_n - y_o)\left(\frac{5}{384}\left\|f^{(4,0)}\right\| h_x^4 + \frac{81}{64}\left\|f^{(2,2)}\right\| h_x^2 h_y^2 + \frac{5}{384}\left\|f^{(0,4)}\right\| h_y^4\right.$$

$$\left. + 16 T_{uv} \left\|A^{-1}\right\|_m \left\|\delta F\right\|\right) \quad \ldots\ldots\ldots \quad 5.5.6$$

5.6 This error analysis is mainly of theoretical interest though it must be of considerable value to show that such bounds exist and can be quantified.

It is interesting to observe that the bound 5.5.6 contains the bound 4.7.4 within its terms and leads one to believe that, if the error analysis of section 4 has been correctly assessed, then the method of section 4, applying one dimensional cubic splines to evaluate double integrals, is superior to this method of section 5 where we use the bicubic spline. Other factors, such as computation time, point to this conclusion also.

6. CONCLUSION

All the computations executed using the methods described here gave results which fell within the expected error bounds. This even when those truncation error bounds were close to the limit of round off error for six digit arithmetic. The difference between the methods was mainly shown to be the computing time that each method used. The Direct Method of approximating to the integral proved to be the speediest. It also gave better results than the other two methods. Using the Direct Method it took only 45 seconds to compute the integral of the function $f(x,y) = \sin x \sin y$ over the region $0 \leqslant x \leqslant \frac{1}{2}\pi$, $0 \leqslant y \leqslant \frac{1}{2}\pi$ with 10 subintervals, yet just about twice as long using the one dimensional cubic splines of section 2. Using the bicubic spline of section 5, with only 2 subintervals, it took approximately 10 minutes to produce a result. However, this result was well within the error bound for that length of subdivision.

In the examples we use only rectangular regions. For the two methods of section 4 we could easily program to apply the methods to more varied regions with only simple restrictions, as laid down in 4.1, but the bicubic spline would have to be restricted in application to regions which can be subdivided by a rectangular mesh.

Some of the examples were computed twice - once using the Lagrange polynomial to estimate boundary derivatives and once using exact boundary derivatives. There were no observable negative effects in the use of the Lagrangian Spline in these examples.

REFERENCES

1.SCHOENBERG, I.J. "Contributions to the problems of approximation of equidistant data by analytic functions."
- Quart.Appl.Math 4 (1946).

2.AHLBERG,NILSON & WALSH "The theory of Splines and their applications"
- Acedemic Press.

3.PRENTER.P. "Splines and Variational Methods"
- Wiley Interscience.

4.DAVIS & RABINOWITZ "Methods of Numerical Integration"
- Acedemic Press.

5.BIRKHOFF & GARABEDIAN "Smooth Surface Interpolation"
-Journal of Maths &Phys (1960) Vol 39,353-368

6.de BOOR C "Bicubic Spline Interpolation"
- J.Maths & Phys (1962) Vol 41, 212-218

7.HALL, C.A "On Error Bounds for Spline Interpolation"
- J.Approx Theory 1 (1968) 209-218

8.CURTIS A.R. "The Approximation of a Function of One Variable by Cubic Splines" in "Numerical Approximation to Functions abd Data" Edited by J.G.Hayes.

9.SWARTZ & VARGA "Error bounds for Spline and L-Spline Interpolation"
J.Approx Theory 6(1972) 6-49

10.CARLSON & HALL "Error bounds for Bicubic Spline Interpolation"
- J.Approx.Theory 7(1973) 41-47.